Comparison of the mean-field approach and simulation in a peer-to-peer botnet case study

Anna Kolesnichenko[1], Anne Remke[1], Pieter-Tjerk de Boer[1], Boudewijn Haverkort[1,2]

[1] Centre for Telematics & Information Technology, University of Twente, Enschede, The Netherlands,
[2]Embedded Systems Institute, Eindhoven, The Netherlands,
{a.v.kolesnichenko, a.k.i.remke, p.t.deboer, b.r.h.m.haverkort}@utwente.nl

Abstract. Peer-to-peer botnets, as exemplified by the Storm Worm and Stuxnet, are a relatively new threat to security on the internet: infected computers automatically search for other computers to be infected, thus spreading the infection rapidly. In a recent paper, such botnets have been modeled using Stochastic Activity Networks, allowing the use of discrete-event simulation to judge strategies for combating their spread. In the present paper, we develop a mean-field model for analyzing botnet behavior and compare it with simulations obtained from the Moebius tool. We show that the mean-field approach provides accurate and orders-of-magnitude faster computation, thus providing very useful insight in spread characteristics and the effectiveness of countermeasures.

Keywords : mean-field approximation, simulation, differential equations, peer-to-peer botnet spread.

1 Introduction

A peer-to-peer botnet can be seen as a very large population (possibly all computers in the Internet) of interacting components (peers), where infected nodes infect more and more other computers. Due to the large numbers of (potentially) active components, the analysis of the spreading speed of such large-scale systems is time consuming and computationally expensive.

Recently, a stochastic model for the growth of peer-to-peer botnets was introduced in [1]. The model is a Stochastic Activity Network (SAN) [2] with an unbounded number of tokens per place, hence, no solutions can be obtained analytically. The authors of [1] simulate the model with the Moebius toolset [3] to gain insight into the effectiveness of defense strategies. Although such simulation is possible, it is very time-consuming and does result in statistical uncertainties.

Recently, much work has been done on the analysis of large populations of interacting objects. Markovian Agents have been used to predict the propagation of earth quake waves [4] or the behavior of sensor networks [5]. The dissemination of gossip information [6] and disease spread between islands [7] was analyzed using mean-field approximation. Hybrid approaches, combining mean-field and simulation, have been proposed for general systems of interacting objects [8], but also to predict predator and prey behavior [9]. Ordinary differential equations (ODEs) have been used to analyze the behavior of intracellular signaling pathways [10] and together with PEPA for epidemiological models [11]. Note that the relationship between the different techniques is currently not well investigated; what all have in common is that they provide an approach to deal efficiently with very large numbers of similar interacting objects.

Out of the many available approaches, in this paper we limit ourselves to mean-field analysis and the direct derivation of ODEs from the SAN. The Markovian agent approach was deemed less suitable here, because it explicitly takes into account locality (i.e., the amount of interaction between entities depends on their location), which is not present in the botnet model studied in [1]. We also have not explicitly tried the approach of deriving ODEs from PEPA, as done in [12]; however, due to similarities among the methods, we expect the resulting ODEs to be the same as the ones we obtained.

While in this paper we are not directly interested in obtaining new insights on botnet behavior, our goal is to show how a quicker analysis method can be used to obtain different measures of interest that cannot be obtained using simulation. We use a model based on the one developed in [1] in order to compare simulation and mean-field approximation. The comparison shows that the mean-field method is much faster than simulation, therefore it allows to address more complicated and resource consuming questions, such as the dependency of botnet spread on several parameters. We explore the speed-up, and show that it can be used it to obtain more insight into the botnet behavior, by taking into account costs for running antimalware software and costs that occur due to computers being infected. Furthermore, we discuss the differences between the mean-Field method and simulation and the resulting suitability in different settings.

For completeness, we point out that the spreading phase of the botnet is quite similar to worm propagation, of which several studies using differential equations have been published. For example, [13] briefly introduces a simple model of worm behavior and compares results with the real measured data, while the main focus of the paper is on the discussion of the different types of worms. The authors of [14] use Interactive Markov Chains to calculate simple bounds of the infected population size and compare results with simulation. A density-dependent Markov jump process model and hybrid deterministic/stochastic model for Random Constant Scanning worm are presented in [15]. A continuous-time approximation of process-algebra models is used in [12] for analysis of the worms spread.

The paper is further organized as follows. In Section 2 we develop a model describing the behavior of an individual computer in a botnet. In Section 3 the mean-field approximation method is applied to the botnet spread model. The mean-field results and the simulation results obtained from Moebius are compared in Section 4. In Section 5 the full potential of the mean-field method in consequence of the attained speedup is explored. In Section 6 the applicability of other large-scale analysis methods to the botnet spread model is discussed. The conclusions and future work are discussed in Section 7.

2 Model of the botnet behavior

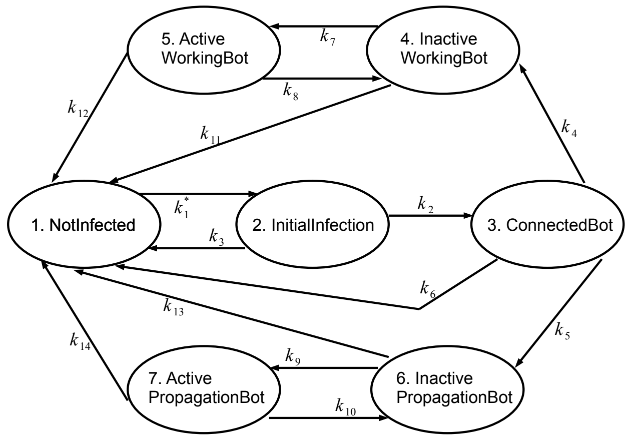

Fig. 1: CTMC for an individual computer.

A Stochastic Activity Network (SAN) model was introduced for Peer-To-Peer botnets in [1]. It models how the infection spreads through an infinite population of computers. The model closely reflects the states a computer goes through after the initial infection has taken place. The original SAN model consist of: (i) one place for each phase of infection a system can be in, that can each hold an unbounded number of tokens, representing the number of computers per phase, and (ii) transitions, which move tokens from place to place, as the infection spreads. The SAN model represents the entire population of infected computers, hence, the number of computers in each state (phase) can be directly derived from the model. However, as the population of computers can be very large or even infinite, it is only possible to derive measures of interest from the SAN model using simulation. This is very time consuming and computationally expensive. Using mean-field analysis, it is possible to compute performance measures of a large population of identical components in a very efficient way. For this, a model is needed which reflects the individual behavior of a single computer. We develop a continuous time Markov chain (CTMC) model for the behavior of an individual computer based on the SAN model from [1], as follows.

Each place in the SAN model is also represented in the CTMC model as an individual state, however, an additional state is added, that represents a computer which is not infected yet. The CTMC model is depicted in Figure 1 and the corresponding transition rates can be found in Table 1.

k_1	RateOfAttack·ProbInstallInitialInfection
k_1^*	Rate depends on k_1 and the environment
k_2	RateConnectBotToPeers·ProbConnectToPeers
k_3	RateConnectBotToPeers·(1-ProbConnectToPeers)
k_4	RateSecondaryInjection·ProbSecondaryInjectionSuccess·(1-ProbPropagationBot)
k_5	RateSecondaryInjection·ProbSecondaryInjectionSuccess·ProbPropagationBot
k_6	RateSecondaryInjection·(1-ProbSecondaryInjectionSuccess)
k_7	RateWorkingBotWakens
k_8	RateWorkingBotSleeps
k_9	RatePropagationBotWakens
k_{10}	RatePropagationBotSleeps
k_{11}	RateInactiveWorkingBotRemoved
k_{12}	RateActiveWorkingBotRemoved
k_{13}	RateInactivePropagationBotRemoved
k_{14}	RateActivePropagationBotRemoved

Table 1: Transition rates for the CTMC of a single computer.

A computer which is not infected yet enters the *InitialInfection* state with rate k_1^* and it becomes initially infected. Then, it connects to the other bots in the botnet, downloads the next part of the malware and possibly moves to state *ConnectedBot* with rate k_2. If the computer for any reason is not able to download the malware it returns to the state *NotInfected* with rate k_3.

After downloading the malware, the computer joins the botnet as either *InactiveWorkingBot* or as *InactivePropagationBot* with rates k_4 and k_5, respectively. If downloading the malware is not possible, for example, because the connection has failed, the computer moves backto the *NotInfected* state with rate k_6. Once the bot becomes either an *InactiveWorkingBot* or an *InactivePropagationBot* it never switches between *Working* or *propagation* bot. In order not to be detected, the bot is inactive most of the time and only becomes active for a very short period of time. Transitions from *InactivePropagationBot* to *ActivePropagationBot* and back occur with rates k_7 and k_8, respectively. The transition rates for moving from *InactiveWorkingBot* to *ActiveWorkingBot* and back are denoted k_9 and k_{10}, respectively.

The computer can recover from its infection, e.g., if an antimalware software discovers the virus, or if the computer is physically disconnected from the network. It then leaves the *InactivePropagationBot* or the *ActivePropagationBot* state and moves to the *NotInfected* state with rates k_{11}, k_{12}, respectively. The same holds for the *working bots*; the transition rates from *InactiveWorkingBot* and *ActiveWorkingBot* are k_{13}, k_{14}, respectively.

The transition rates in the SAN model are marking dependent, i.e., the rate of moving from state s_1 to state s_2 linearly depends on the number of computers in state s_1. The transition rates for the CTMC model of the single computer, as proposed here, are constant, with the only exception of k_1^*, which depends on the number of active propagation bots in the environment; seen from this perspective this CTMC is a non-homogeneous CTMC.

The CTMC model consists of seven states ($S_1, .., S_7$), where each state from the state space $S = (\mathit{NotInfected}, ..., \mathit{ActiveWorkingBot})$, $|S| = 7$, represents a certain phase of the infection of a single computer. The rate matrix R of the CTMC is written as:

$$R = \begin{pmatrix} 0 & k_1^* & 0 & 0 & 0 & 0 & 0 \\ k_3 & 0 & k_2 & 0 & 0 & 0 & 0 \\ k_6 & 0 & 0 & k_4 & 0 & k_5 & 0 \\ k_{11} & 0 & 0 & 0 & k_7 & 0 & 0 \\ k_{12} & 0 & 0 & k_8 & 0 & 0 & 0 \\ k_{13} & 0 & 0 & 0 & 0 & 0 & k_9 \\ k_{14} & 0 & 0 & 0 & 0 & k_{10} & 0 \end{pmatrix} \tag{1}$$

The $|S| \times |S|$ generator matrix Q is given as follows: $Q_{s_1 s_2}$ is equal to the transition rate $R_{s_1 s_2}$ to move from the state s_1 to the state s_2 and Q_{ss} is equal to the negative sum of all the rates in row s. The only exception is the rate k_1^* to move from the *NotInfected* state (S_1) to the *InitialInfection* state (S_2). As discussed above, this rate depends on k_1 and on the number of computers in the *ActivePropagationBot* state, but it should not depend on the total number of computers in the environment. We provide an expression for k_1^* in the next section.

In the following, we use the mean-field method to model and study the population of computers in a network that can possibly be infected, assuming that all computers behave individually according to the CTMC model described above. We leave the discussion of actual parameter values to Section 4.

3 Mean-field approximation

The mean-field method allows to compute the exact limiting behavior of an infinite population of identical components, and suggests an approximation when the number of components is sufficiently large. The global idea of the method is to describe the behavior of the infinitely large population (overall behavior) via the average behavior of the individual components, which are indistinguishable. Both the overall behavior of the population and the individual behavior of a single component are modeled as a Markov chain, where the transition rates in the overall Markov model depend on the number of components in each state, but not on the total number of components in the system.

Previously we discussed the CTMC (see Fig. 1) and the corresponding state space S which describes the behavior of a single computer in the botnet. The overall behavior of the population of N computers is then given by a CTMC with a state space of size $|S|^N$. This can be lumped due to the identicality of the computers' behavior, and then described by the number of computers in each state $s \in S$ at time t, i.e., by $\underline{M}(t) = (M_1(t), M_2(t)...M_{|S|}(t))$, where $M_s(t)$ is the number of computers in state s.

Recall that the generator matrix $\mathbf{Q}$, as discussed in the previous section, has one transition rate that depends on the environment: k_1^*. We can now express this k_1^* in terms of $M(t)$ from the overall CTMC, as follows. The total rate of infections produced by all bots that are in the active propagation state is $k_1 \cdot M_7(t)$. These infections are spread out randomly over all not-yet infected computers, of which there are $M_1(t)$. Hence, the infection rate k_1^* perceived by each individual computer is given by the ratio:

$$k_1^*(\underline{M}(t)) = \frac{k_1 \cdot M_7(t)}{M_1(t)}. \tag{2}$$

The above equation completes the definition of $\mathbf{Q}$ and, hence, allows to build a complete mean-field model. Given a population of N computers, we denote the fraction of objects in an arbitrary state s at time t as $m_s(t) = M_s(t)/N$, where $0 \leq \underline{m}(t) \leq 1$ is the normalized occupancy vector $\underline{m}(t) = (m_1(t), m_2(t)..., m_{|S|}(t))$, which does not depend on N. The generator matrix $\mathbf{Q}(\underline{m}(t))$ for the normalized vector $\underline{m}(t)$ is the same as $\mathbf{Q}(\underline{M}(t)$ for the unnormalized vector, since its components depend only the ratios of the vector elements.

We apply the mean-field method for the overall system using Theorem 1 from [16], which states that the normalized occupancy vector $\underline{m}(t)$ of the overall behavior tends to be deterministic in distribution and satisfies the following differential equations when N tends to infinity:

$$\frac{d\underline{m}(t)}{dt} = \underline{m}(t)\mathbf{Q}(\underline{m}(t)). \tag{3}$$

The system of equations describing the population behavior in the botnet then equals:

$$\begin{cases} \dot{m}_1(t) = k_3 m_2(t) + k_6 m_3(t) + k_{11} m_4(t) \\ \qquad\qquad +k_{12} m_5(t) + k_{13} m_6(t) + (k_{14} - k_1) m_7(t), \\ \dot{m}_2(t) = -(k_2 + k_3) m_2(t) + k_1 m_7(t), \\ \dot{m}_3(t) = k_2 m_2(t) - (k_4 + k_5 + k_6) m_3(t), \\ \dot{m}_4(t) = k_4 m_3(t) - (k_7 + k_{11}) m_4(t) + k_8 m_5(t), \\ \dot{m}_5(t) = k_7 m_4(t) - (k_8 + k_{12}) m_5(t), \\ \dot{m}_6(t) = k_5 m_3(t) - (k_9 + k_{13}) m_6(t) + k_{10} m_7(t), \\ \dot{m}_7(t) = k_9 m_6(t) - (k_{10} + k_{14}) m_7(t). \end{cases} \tag{4}$$

Note that in the above, k_1^* is used as in (2), where the entries of the unnormalized vector $\underline{M}(t)$ have been replaced with the corresponding entries of the normalized occupancy vector $\underline{m}(t)$. These equations can be solved analytically, however the closed forms are impractically large. We use the Wolfram Mathematica tool to obtain the analytical solution. The system of ODEs (4) is applicable when N tends to infinity. To obtain the approximation for the case when N is finite but sufficiently large, we use the Corollary 2 from [16], which states, that:

When N is sufficiently large, the normalized state vector of the lumped process, $\underline{m}(t)$, is a random vector whose mean can be approximated by the following differential equation

Parameter	Baseline experiment	Experiments 1	2	3	4	5	6
ProbInstallInitialInfection	0.1	**0.06**	**0.04**	0.1	0.1	0.1	0.1
ProbConnectToPeers	1	1	1	1	1	1	1
ProbSecondaryInjectionSuccess	1	1	1	1	1	1	1
ProbPropagationBot	0.1	0.1	0.1	0.1	0.1	0.1	0.1
RateOfAttack	10.0	10.0	10.0	10.0	10.0	10.0	10.0
RateConnectBotToPeers	12.0	12.0	12.0	12.0	12.0	12.0	12.0
RateSecondaryInjection	14.0	14.0	14.0	14.0	14.0	14.0	14.0
RateWorkingBotWakens	0.001	0.001	0.001	0.001	0.001	0.001	0.001
RateWorkingBotSleeps	0.1	0.1	0.1	0.1	0.1	0.1	0.1
RatePropagationBotWakens	0.001	0.001	0.001	0.001	0.001	0.001	0.001
RatePropagationBotSleeps	0.1	0.1	0.1	0.1	0.1	0.1	0.1
RateInactiveWorkingBotRemoved	0.0001	0.0001	0.0001	0.001	0.001	0.001	0.001
RateActiveWorkingBotRemoved	0.01	0.01	0.01	0.01	0.01	0.01	0.01
RateInactivePropagationBotRemoved	0.0001	0.0001	0.0001	0.001	0.001	0.001	0.001
RateActivePropagationBotRemoved	0.01	0.01	0.01	**0.07**	**0.04**	**0.02**	**0.015**

Table 2: The setups for the different experiments. Bold font indicates difference w.r.t. baseline experiment.

$$\frac{dE(m_i(t))}{dt} \approx \sum_{j \in S} E(m_j(t)) Q_{ji}(\underline{m}(t)), i \in S. \qquad (5)$$

Considering this, the expected occupancy vector $E(\underline{M}(t))$ is given as follows: $E(\underline{M}(t)) \approx N \cdot \underline{m}(t)$, where $\underline{m}(t)$ is the solution of (4). In our experiments we set $N = 10^7$.

4 Results

In this section we discuss the mean-field results in detail and compare them to the simulation results we obtained from the model given in [1]. We carred out out a similar series of experiments as in [1]; the chosen parameters for all these experiments are given in Table 2.

The simulation of the model was done using the Moebius tool [3]. Each experiment covered one week of simulated time. Each experiment was replicated 1000 times; the mean values and 95% confidence intervals of the measures of interest are shown. The initial conditions for each experiment are as follows: 200 computers are located in the place *ActivePropagationBots* in the SAN, and all the other places are empty. Note that the simulation results shown here differ from those in [1]. Together with the authors of [1] we found a small mistake in the simulator settings they used: because the rates in the SAN model are marking dependent, a flag has to be set in the Moebius tool to ensure that the rates are updated frequently. Not setting this flag can result in inaccurate numbers of propagation bots, as illustrated below in Figure 2.

We use Mathematica [17] to obtain solutions for the set of differential equations (4) coupled with the transition rates from Table 2. To obtain the same initial conditions for the mean-field model as for the SAN model we need to take the additional state in the CTMC into account. Given an overall population of $N = 10^7$, the fraction of computers in the state *NotInfected* is initialized as $m_1(0) = (N-200)/N$, the fraction of computers in the state *ActivePropagation-Bot* is initialized as $m_7(0) = 200/N$, and the fractions of computers in all other states are initialized as zero.

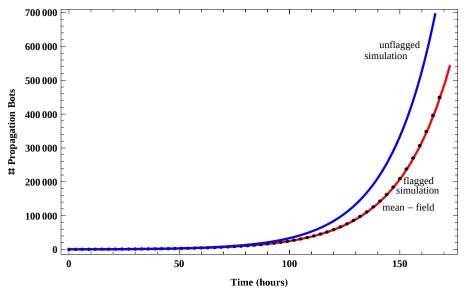

Fig. 2: Number of propagation bots over time in the Baseline experiment obtained from Moebius simulations with and without the rates-updating flag set, as well as obtained from mean-field approximations.

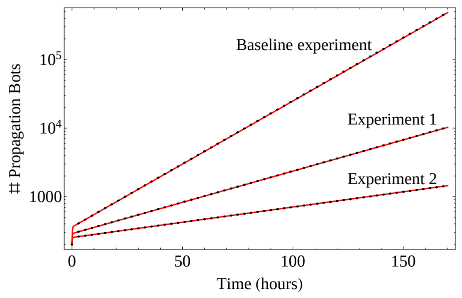

Fig. 3: Number of propagation bots over time in the Baseline experiment and in Experiment 1 and 2 obtained from Moebius simulation and from mean-field analysis

Figure 2 shows the number of the propagation bots in a botnet. The number of propagation bots (both active and inactive, states S_6 and S_7) has been taken as measure of interest since they actively infect "healthy" computers. The lower solid line depicts the mean-field results of the Baseline experiment together with the 95% confidence intervals of the corrected Moebius simulation. As can be seen, the mean-field results are very accurate in this case, since they lie mostly within the confidence intervals, even though the confidence intervals are very narrow. The upper solid line represents the mean value of the original Moebius simulation from [1]. Comparing the original Moebius results with the new results from the correct simulator setting, reveals that the number of propagation bots (both *active* and *inactive*) differs from the results stated in [1]; during the first fifty hours the unflagged simulation provides slightly lower results (about 20%), however on this scale the difference is hardly noticeable. Starting from fifty hours, the unflagged simulation over-estimates; after a week the difference is about 42% (754755 vs. 440073). Note that the simulation takes longer than 5 days of runtime, versus 1 second for the mean-field method. We come back to this at the end of the section.

The goal of this paper is not to study the growth of botnets under different conditions, but to compare the results obtained from mean-field approximation with those obtained from simulations. Hence, we compare results for a representative selection of experiments in order to discuss the advantages and disadvantages of both approaches.

To investigate how a reduced infection spread would influence the growth of botnets, Experiments 1 and 2 were done in [1]. The "user factor" (*ProbInstalInfection*) is reduced to 60% and 40%, respectively, as compared to the Baseline experiment to represent a lower probability of, e.g., opening infected files. The results are, together with those from the Baseline experiment, presented in Figure 3. A logarithmic scale has been chosen for the number of propagation bots, in order to better visualize the exponential growth. For both experiments, the results obtained with the mean-field model are very accurate and lie well within the confidence intervals most of the time.

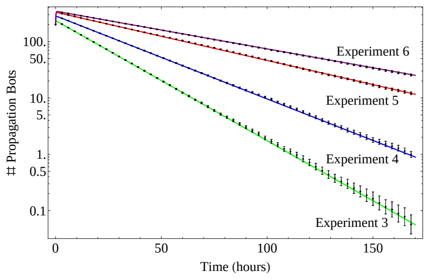

Fig. 4: Number of propagation bots over time in Experiments 5, 6, 7, and 8, obtained from Moebius simulation and from mean-field analysis.

Experiment	Simulation	mean-field
Baseline	5 d., 3 h., 25 min.	1 sec.
Experiment 1	9 h., 51 min.	1 sec.
Experiment 2	5 h., 37 min.	1 sec.
Experiment 3	31 min	1 sec.
Experiment 4	40 min	1 sec.
Experiment 5	45 min	1 sec.
Experiment 6	36 min	1 sec.

Table 3: Time spent on simulation and mean-field approximation.

To investigate how efficient anti-malware software can control or even stop the botnet spread, experiments with increased removal rates were done in [1]. To make a comparison of the approaches, we conducted a series of experiments, where the removal rate of active propagation bots varies between 0.01 and 0.1. The mean-field approximation provides an explicit result, which in most of the cases lies well within the 95% confidence intervals (see Figure 4).

At first sight the high accuracy of the analytical results might be surprising, since the underlying assumption of mean-field approximation is that the number of interacting components is large. However, apparently in Experiment 3 (cf. Figure 4) the initial set of active propagation bots hardly gets a chance to infect more computers before being disinfected themselves. In terms of the local CTMC, it means that the transition from the state *NotInfected* to the state *InitialInfection* is taken by (almost) none of the computers. This transition happens to be the only one whose rate depends on the environment; if we remove it from the local CTMC, we are left with a CTMC with constant rates. With all rates in the CTMC constant, the ODEs (4) are easily seen to be the Kolmogorov differential equations, whose solution is the probability distribution over the states of the CTMC as a function of time. Also, removing this transition in the SAN simulation model reduces it to a set of many *independent* CTMCs. Taking the number of markings per state as a function of time, and dividing by the total, obviously results in an unbiased estimate of the probability distribution of the CTMC in the course of time. Thus, clearly the two approaches should match, as they in fact do and this explains why the mean-field results are still accurate, even though in this case the overall number of components is small.

It is interesting that the confidence intervals in Experiment 6 are much narrower than the ones in Experiment 3. As the average number of propagation bots decreases over time, the confidence intervals seem to get wider on the logarithmic scale (see Figure 4). In fact, however, the absolute width of the intervals gets smaller, but less quickly than the estimate itself. The reason for this is that the actual number of propagation bots always is a non-negative integer; therefore, when the estimated *average* decreases much below 1, it must be the average of many 0's and a few 1's (or even fewer higher integers). Such an estimate inherently has a large coefficient of variation; in fact, this is the main problem of *rare-event simulation*, cf. [18].

Another thing to remark about these experiments, is that when the number of propagation bots reaches 0, and there are also no bots in the states *InitialInfection* and *ConnectedBot* anymore, no new infections can occur. The number of propagation bots will then remain 0. Thus, when the graph indicates that after a week the *average* number of propagation bots is about 0.01, this means that in most (about 99%) of the simulation runs the botnet is extinct and will stay so, while only a few runs still have some botnet activity.

In Table 3, the computer run times for the simulation and for the mean-field computation are compared. The results were obtained on a Core-i7 processor with 3 GB RAM and 4 cores and hyper threading. One sees that the simulation can take a very long time, namely up to several days, while the mean-field approximation is always done within one second. The difference between the simulation time for the different experiments is due to the marking dependency of the rates. For example, in the Baseline experiment the number of *ActivePropagationBots* is large, hence, the rate of infection becomes very large and more time is needed to simulate the resulting large number of events. The time spent on the simulation of the experiments with lower numbers of computers involved is reasonably smaller; however the mean-field approximation is still much faster in all cases. In any case, the simulation times should only be taken as indications, since the simulations were not run completely independently, but with 2, 3 or 4 of them simultaneously on a 4-core computer, so the jobs may have interfered with each other.

5 Exploiting the speed-up

In the previous section we have shown how fast and efficient the mean-field method is (cf. Table 3), and that it gives correct results. This allows us to use the mean field method in this section to address problems which are not feasible using simulation: (i) we study the dependence of the botnet spread on two parameters, while the results in the previous section are only functions of time for a given set of parameter values, (ii) and we study the behavior of the botnet in the presence of cost constraints. The purpose of this section is to show the difference between the simulation and the mean-field capabilities, and, at the same time, to show the advantages of the fast analysis.

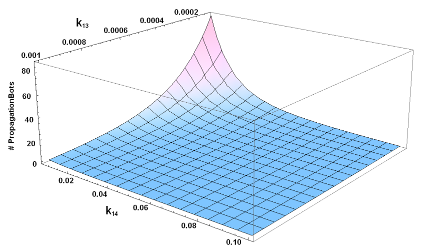

Fig. 5: Number of propagation bots for $(k_{13}, k_{14}) \in [8 \cdot 10^{-5}; 10^{-3}] \times [8 \cdot 10^{-3}; 10^{-1}]$ at time $T = 3days$, all other parameters are the same as for Baseline experiment (see Table 2).

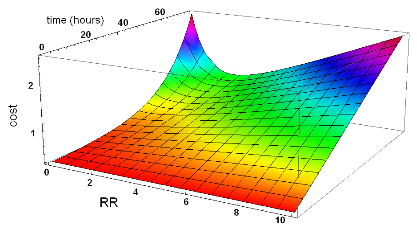

Fig. 6: Cost of the system performance for $D_1 = 0.01, D_2 = 4 \cdot 10^{-5}$.

The authors of [1] used time-consuming simulation to show in a couple of examples that there is no considerable difference in increasing the detection of active or inactive bots (namely increasing the removal rates k_{11}, k_{13} or k_{12}, k_{14}). The mean-field method allows to make the analysis faster and to obtain more information. We calculate the number of propagation bots as a function of k_{13} and k_{14} (see Figure 5). As one can see, there is no considerable difference in a relative increase of one or the other parameter. It is known that inactive computers are much harder to detect (increasing k_{13} is more difficult), therefore the above results might be helpful for the antivirus software developers to find the better strategy for botnet removal.

Next, we introduce a cost concept to analyze the economical side of an infection. In the following, two types of costs are considered: (i) the cost of a computer being infected, that occurs for example due to the loss of information or productivity, and (ii) the cost of more frequent checking with antivirus software. On one hand the number of infected computers, and hence their cost grows if computers are not frequently checked. On the other hand, if computers are checked too often the botnet is not growing, but running the antivirus software becomes very expensive. We analyze this trade-off in more detail in the following. We calculate the cumulative cost as follows:

$$C(t_0, t_1, RR, D_1, D_2) = \int_{t_0}^{t_1} (D_1 \cdot \text{InfBots}(t, RR) + D_2 \cdot RR \cdot \text{AllComp}) \, dt \tag{6}$$

where RR is the change in removal rates $k_{11}, ..., k_{14}$ with respect to the rates in the Baseline experiment, i.e. $k_{11} = RR \cdot k_{11,baseline}$ and similarly for k_{12}, k_{13}, k_{14}; D_1 is the cost of infection; InfBots(t, RR) is the number of infected computers for a given RR, at time t, including active and inactive working and propagation bots; D_2 is the cost of one computer being checked, which probably is much lower than the cost of infection (D1); AllComp is the number of the computers in the system.

We calculate the cumulative cost of the system performance for three days. For RR from the interval $[0.001; 5]$ we calculate the cost as a function of time for given D_1 and D_2. Results are depicted in Figure 6 and, one can see, that the cost grows exponentially with time and quite linearly with decreasing RR if the computers are not checked frequently (for the RR between 0 and 1). However, if antimalware software is used too often (RR above 2), the cost grows linearly with RR.

We see that the mean-field method can be easily used for finding the removal rates which minimize the cost at a given moment of time. It can help network managers with careful decision-making, based on the situation at hand. Even though not all parameters might be known in reality, such analysis can help to obtain a better understanding of the characteristics of botnet spread.

In this section we studied different aspects of botnet spread and gained a deeper understanding of the trade-off which occurs when costs are induced. The set of problems discussed in this section demonstrates the efficient application of the mean-field method. One can think of other questions to address, however our aim was to show the potential of the method by addressing problems which can not be solved using simulation.

6 Variations of the method

As an alternative to the method described in Section 3, we also applied a discrete-time approximation to the model. The uniformisation of the CTMC was done and the corresponding DTMC for the single node behavior was obtained. We used the mean-field convergence theorem from [6] to obtain the approximation. As was expected, the results for discrete time approximation are identical to the results for the continuous time approximation, and therefore omitted here.

In [10], a method is proposed to systematically derive a set of ordinary differential equations governing the concentrations of reactants in (bio)chemical systems. This approach can also be applied to the botnet model, by interpreting each of the 7 states as a chemical "reactant", and each transition between the states as a "chemical reaction". The concentrations then represent the fraction of all computers that are in that particular state; the method allows sufficient freedom in the dependence of the reaction rates on the concentrations to fit the botnet model. Applying the systematic method from [10] to this model results in a set of ODEs that is identical to the equations (4) which we derived using the mean-field approximation.

It turns out that there is a good explanation for the match between this ODE method and the mean-field approximation. The main premise in mean-field analysis is that there is a large number N of identical entities (computers in the botnet case), of which some number are in each state. The quantity of interest then is the *fraction* of the entities that is in each state (the vector $\underline{m}(t)$). Now, a *concentration* in the (bio)chemical context of [10] is also a fraction, namely the fraction of all molecules that are of a certain type (assuming that the number of solvent molecules far exceeds the others, so that the total number of molecules is practically constant). With this interpretation, the two methods are identical. The main difference between the two methods is that in mean-field analysis, the total number of entities is called N and is explicitly taken in the limit to infinity. In contrast, this limit is not made explicit in the (bio)chemistry ODE method; this may be justified by the typically extremely large number of molecules in chemical systems (cf. Avogadro's number).

A similar argument holds for ODEs derived from a PEPA model, as done in [12]. The PEPA model describes a large CTMC, which in fact models many identical computers, each of which can be in one of a small number of states. For deriving ODEs, only the total number per state is taken into account. These total numbers are then treated as continuous rather than discrete variables, which is equivalent to setting the total number to infinity. Although we did not explicitly do so, it is clear that this PEPA-based approach to our botnet model would again result in the same set of ODEs.

7 Conclusion

In this paper, we have compared different approaches for evaluating a Markov model for peer-to-peer botnet spreading. We have shown that the mean-field approach is much faster than simulation, taking about 1 second instead of minutes to days of computation time. The results from the mean-field analysis match those from the simulation very well, being mostly inside the 95% confidence interval.

Due to the speed-up of the mean-field method we have been able to address various questions which cannot practically be answered with simulation, such as questions involving cost trade-offs; this is useful in typical engineering applications. We also considered several other approaches that can be used for the analysis of large-scale systems, such as automatically deriving ODEs and deriving ODEs from process algebra models, and found them to be equivalent to the mean-field approach.

In general, the mean-field method is only a first-order approximation to the real Markov chain model, which becomes better as the number of entities involved increases. However, in the present model we did not observe any significant difference between the mean-field results and the simulation results. In contrast to the mean-field approximation, the precision of the simulation results suffers when the mean number of bots being estimated, becomes close to zero.

This is because the standard deviation does not go to zero as fast as the mean value. Note that this is, in fact, the problem that *rare-event simulation* addresses.

The present research shows the usefulness of the mean-field approach, as it is able to provide very accurate results very quickly. However, even in cases where mean-field results are less accurate for small population numbers, it can be useful as a quick check of the simulation. In fact, the simulator setting problem discussed in Section 4 was found due to the mismatch with the mean-field results.

Future work will involve further exploration of the conditions under which mean-field results are correct. As noted above, even when the number of entities involved was small, our mean-field results remained correct, and we could explain why this was the case. Presumably, more general conditions for such correctness can be found.

Acknowledgments

This work is supported by the Netherlands Organization for Scientific Research (NWO), the Centre for Telematics and Information Technology (CTIT), and the Centre for Dependable ICT Systems (CeDICT). Furthermore, we thank the authors of [1] for providing the source code of their model and discussions on the settings of the simulator.

References

[1] van Ruitenbeek, E., Sanders, W.H.: Modeling peer-to-peer botnets. In: 5th Int. Conference on Quantitative Evaluation of SysTems, (QEST'08), IEEE CS Press (2008) 307–316

[2] Sanders, W., Meyer, J.: Stochastic Activity Networks: Formal Definitions and Concepts? Lectures on Formal Methods and PerformanceAnalysis (2001) 315–343

[3] Deavours, D., Clark, G., Courtney, T., Daly, D., Derisavi, S., Doyle, J., Sanders, W., Webster, P.: The Mobius framework and its implementation. Software Engineering, IEEE Transactions on **28**(10) (2002) 956–969

[4] Cerotti, D., Gribaudo, M., Bobbio, A.: Disaster Propagation in Heterogeneous Media via Markovian Agents. Critical Information Infrastructure Security (2009) 328–335

[5] Gribaudo, M., Cerotti, D., Bobbie, A.: Analysis of on-off policies in sensor networks using interacting markovian agents. In: Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on, IEEE (2008) 300–305

[6] Bakhshi, R., Cloth, L., Fokkink, W., Haverkort, B.: Mean-Field Analysis for the Evaluation of Gossip Protocols. In: 6th Int. Conference on Quantitative Evaluation of SysTems, (QEST'09), IEEE CS Press (2009) 247–256

[7] Bakhshi, R., Endrullis, J., Endrullis, S., Fokkink, W., Haverkort, B.: Automating the mean-field method for large dynamic gossip networks. In: 7th Int. Conference on Quantitative Evaluation of SysTems, (QEST'10), IEEE CS Press (2010)

[8] Le Boudec, J.Y., McDonald, D., Mundinger, J.: A generic mean field convergence result for systems of interacting objects. In: 4th Int. Conference on Quantitative Evaluation of SysTems, (QEST'07), IEEE CS Press (2007) 3–18

[9] Henzinger, T., Mateescu, M., Mikeev, L., Wolf, V.: Hybrid Numerical Solution of the Chemical Master Equation. In: Proceedings of Computational Methods in Systems Biology (CMSB 2010). (2010) preprint arXiv:1005.0747.

[10] Calder, M., Gilmore, S., Hillston, J.: Automatically deriving ODEs from process algebra models of signalling pathways. In: Proceedings of Computational Methods in Systems Biology (CMSB 2005). (2005) 204–215

[11] Ciocchetta, F., Hillston, J.: Bio-PEPA for epidemiological models. Electronic Notes in Theoretical Computer Science **261** (2010) 43–69

[12] Bradley, J., Gilmore, S., Hillston, J.: Analysing distributed internet worm attacks using continuous state-space approximation of process algebra models. Journal of Computer and System Sciences **74**(6) (2008) 1013–1032

[13] Feamster, N., Gao, L., Rexford, J.: How to lease the internet in your spare time. SIGCOMM Comput. Commun. Rev. **37** (2007) 61–64

[14] Garetto, M., Gong, W., Towsley, D.: Modeling malware spreading dynamics. In: INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies. Volume 3. (2003) 1869 – 1879 vol.3

[15] Rohloff, K., Basar, T.: Stochastic behavior of random constant scanning worms. In: Computer Communications and Networks, 2005. ICCCN 2005. Proceedings. 14th International Conference on. (2005) 339 – 344

[16] Gribaudo, M.: Analysis of large populations of interacting objects with mean field and markovian agents. In: Computer Performance Engineering. Volume 5652 of Lecture Notes in Computer Science. Springer Berlin (2009) 218–219

[17] Wolfram Research, Inc.: Mathematica tutorial. `http://reference.wolfram.com/mathematica/tutorial/IntroductionToManipulate.html` (2010)

[18] Heidelberger, P.: Fast simulation of rare events in queueing and reliability models. ACM Transactions on Modeling and Computer Simulation **5** (1995) 43–85